SAMSUNG Knox

Whitepaper:

Samsung Knox™ Security Solution

Version 2.2 May, 2017

Samsung Research America

Samsung Electronics Co., Ltd.

Contents

Section 1: BYOD and mobile security

Personal smartphones and mobile devices permit employee access to corporate email and network resources, but add a vulnerable entry point that can be exploited. Additionally, document sharing enables mobile devices to further proliferate corporate resources outside of a potentially compromised network infrastructure. The evolution of *Bring-Your-Own-Device* (BYOD) and *Corporate-Owned-Personally-Enabled* (COPE) strategies started slowly, and then accelerated with the proliferation of apps for every business and personal need. While enterprise employees enjoyed the freedom and productivity of continuous connectivity, IT admins on the other hand were blind-sided with protecting corporate owned devices from the massive amounts of insecure personal data employees began keeping on their phones.

Legacy enterprise IT admin security models were designed to protect the integrity of the enterprise network and company-issued PCs, not the personal smartphones and tablets utilized by the enterprise's employees. With both BYOD/COPE and cyber-attacks increasing, the scramble to analyze the facts and figures ensued in hopes of finding a way to manage the escalating problem and complexity of mobile device security.

What are the numbers? What are the risks?

Mobile malware attacks increased 300% between 2015 and 2016 according to a new 2017 issued report from Kaspersky labs

In 2016, the number of reported malware cyber-attacks was more than 8.5 million; that's three times more than reported in 2015, according to a Kaspersky Lab report on mobile device malware growth.

Additionally, Kaspersky registered almost 40 million attacks by malicious mobile software over the course of the 2016 as well.[1]

IT admins must quickly identify potential threats and understand how the threat profile is growing, since mobile malware attacks increased more than 300% between 2015 and 2016.[2]

Malicious and poorly designed mobile device applications aren't the only security threat in the mobile landscape, but they are the biggest threat. A Nielsen February 2014 report, The Digital Consumer, reported that smartphone owners spend 86% of their time using apps versus the mobile web.[3] Consequently, the real culprit for poor mobile security is the open source code hackers can easily use to create and distribute malicious apps.

Even while these figures were being reported, Samsung was already at work designing a solution for mobile devices. In 2012, a group of Samsung engineers built a new mobile environment, deeply rooted in the hardware of the Android *operating system* (OS) that could be used for any other mobile OS The blueprint for this solution included maintaining a highly trusted platform, and the tools required for an enterprise-ready security solution. In 2013, Samsung released Samsung Knox (TM) for any size enterprise, affording them complete control over how they implement their mobile security model.

Section 2: Background: What’s in a smartphone?

There is much more to a smartphone than the apps and widgets users typically experience. Behind the mobile device interface exists a sophisticated system of advanced processor architectures, operating system kernel, libraries, middleware, and security services.

Smartphone hardware

The *processor* is a smartphone's central computational unit where its apps and OS reside and run. The processor is physically connected to the phone’s antennas, internal storage drives, removable SD cards, and docking ports.

The modes of execution represent one of the most important features of a processor. A *mode* defines how much privilege a piece of software running on the processor has been granted. For example, when user-installed apps run on the processor, the processor runs in *user mode*. In user mode, the apps are not allowed to directly access hardware devices or resources controlled by other apps. On the other hand, when critical operating system software is running, the processor is in *privileged mode*. In privileged mode, the system’s software can directly access hardware devices, as well as all data held by the user’s applications. Any code running in privileged mode must be protected from control by adversaries wishing to exploit the device.

Knox leverages a processor architecture known as ARM® TrustZone® . While TrustZone maintains the two modes described above, it also provides a new security-specific construct called *worlds*. In TrustZone, there are two worlds, the *Normal World*, and the *Secure World*. Virtually all smartphone software as we know it today still runs in the Normal World. The Secure World is reserved for highly-sensitive computations, such as those involving *cryptographic keys*. Knox utilizes the TrustZone's Secure World extensively for protecting enterprise confidential data and monitoring the OS kernel running in the Normal World.

The Android operating system

A smartphone's hardware processor utilizes both a user and privileged mode for various functions. The portion running in privileged mode is called the *kernel.* OS kernels are among the most rigorously engineered pieces of software in the world, since they must perform many functions, all with the power of the processor's privileged mode. For example, any time phone data arrives from the Internet, the OS kernel first chooses whether to even allow the data to proceed, or to drop it. If the data is allowed, the kernel examines it and decides which application the data is intended. The kernel then places the data in the app's memory, and notifies the app data has arrived. If the app wishes to send a reply, the app's reply is sent by repeating this whole process in reverse.

Given this example, consider what could happen if an attacker gained control of the OS kernel. Due to the kernel's high permissions, the attacker could leak arbitrary sensitive data from any application, and send it anywhere on the Internet. This is a compelling reason why Knox implements its extensive protections for OS kernels.

Inter-application processing

When applications communicate with one another they ask the kernel to establish lines of communication. To facilitate more robust app communication, the Android OS provides another layer of software, called *middleware*.

The Android middleware runs in user mode. The middleware provides rich methods that allow apps to share their data and perform operations on each other's behalf. For example, many image library apps can take photos, even though they don't know how to use the phone's camera. Instead, they simply request that an app that does understand the camera take the picture on their behalf. A second major function is ensuring such communication does not occur between enterprise apps and user apps. This prevents sensitive data from leaking to an untrusted third-party, and prevents corrupted data from entering enterprise apps.

Boot process

The boot process binds the hardware, kernel, and apps. When a device is initially turned on, the user's applications are not immediately available. Instead, a succession of software components start, with each component starting the next one in the chain. Typically, when the user presses the ON button, the device first runs a program called a *bootloader*. Many mobile device architectures use multiple bootloaders to perform different functions. The bootloader then finds where the kernel is stored, and begins running the kernel in the processor's privileged mode. The kernel starts the Android middleware and some basic apps, running them in user mode. Once the boot completes, the user is queued to login into their phone.

Mobile device security

Mobile security is quite different from other domains, in that device owners have complete control over how to use and secure their own devices, as opposed to a corporate-owned laptop, controlled by IT admins. With COPE and *Corporately Owned, Business Only* (COBO), sensitive emails are likely downloaded to the device, but the user may simultaneously compromise the kernel's security to allow for device customizations. The process of unlocking a device's operating system so you can install unapproved apps is typically known as device *rooting*.

The security breach inherent in device rooting makes it easier for malicious parties to exploit the rooted device. The same techniques are known by malware authors, and can enable them to steal the user's data or attack different targets. In response, many of the security measures implemented by Knox are designed to either prevent rooting, or mitigate the resulting damage.

Cryptography

Cryptography is another fundamental measure for mobile device security. Knox uses cryptography for three key functions:

- *Encryption* - the random re-arrangement of data using a protected key
- *Hashing* - the creation of a unique series of numbers to represent a particular piece of software or data; a single difference in a piece of data yields a different hash
- *Signing* - the encryption of hashed data using a private key to prove the data originated from a particular known entity

Knox frequently uses signing to produce the hash signatures of firmware components. This proves the firmware component originated from the owner of the private key used for signing, and proves the component originated from Samsung. Knox signing keys are only accessible in the TrustZone Secure World.

Section 3: Samsung Knox overview

Enterprise data is finding its way on employee smartphones as the new norm for corporate efficiency. Corporate smartphones have increased productivity by placing work and personal data on the same device, but has compounded enterprise security exponentially. Mobile devices expose numerous pathways through which sensitive data can be stolen (sharing data with untrusted third-party applications, device theft, intentional rooting, misconfigured, or vulnerable enterprise applications). These vulnerabilities grow exponentially across an organization, when hundreds or even thousands of devices require secure management, configuration, and deployment.

Knox is the most comprehensively secure and manageable mobile device solution for enterprises large and small. Based on the Android OS, Samsung Knox is designed on the philosophy that device security should be rooted in fixed hardware mechanisms. Knox bases this foundation in the principles of *trusted computing*, a set of methods for making devices that can prove to enterprises they are running the correct security software, and can raise alerts when tampered. On top of this more stable foundation, Knox builds a workspace environment to protect enterprise apps and their data, a robust set of data at rest protections, and a large suite of enterprise security tools, including a highly configurable *Virtual Private Network* (VPN), *Single Sign On* (SSO) and *Enterprise Mobility Management* (EMM) interfaces.

The Samsung Knox philosophy

Samsung designed Knox using a industry leading two-step design philosophy:

Step 1.Build a trusted environment rooted in proven hardware security mechanisms.

In a trusted environment, sensitive or enterprise-critical functionality is only enabled once the device is in an allowed state. In this context, *state* refers to the security-relevant software and configurations on the device. For example, parts of state considered by Knox include the bootloaders, kernel, TrustZone, and numerous security policy configurations. Furthermore, the trusted environment allows for *remote attestation*, a process where any change is securely validated by a set of proofs. Third parties can then inspect these proofs to decide if the device state meets their security requirements. A device's trusted environment is rooted in hardware when its security is based on the prevention of physical tampering. Why is it important to root trust in device hardware? Knox designers are aware operating system security has historically been trusted to privileged system software (mainly the OS kernel). However, in the last two decades, attackers have become more successful at exploiting kernel flaws. Other mechanisms such as heavyweight virtual machines or special *Basic Input/Output System* (BIOS) checks have been implemented and circumvented. The Knox design recognizes the single best defense against a full-system compromise is to tie system self-checks to a secret password maintained by secure hardware and out of the reach of any software-based or physically present adversary.

Step 2. Make the trusted platform ready for enterprise use

Enterprises require a trusted platform for their critical data security. Such accessibility involves giving enterprises complete control over their data. Knox includes a collection of useful secure applications and utilities that enable enterprise-ready deployment. Knox Workspace security is based in the hardware root of trust and isolating the workspace from the personal space.

Samsung Knox design

Knox addresses the most pressing security problems facing enterprises' COPE and COBO strategies today. Samsung has identified the following key challenges in making an Android-based system enterprise ready:

- Device rooting
- The mixing of enterprise data with user apps on the same device
- Device theft
- The difficulty of securing custom enterprise applications
- The lack of enterprise manageability and supporting utilities

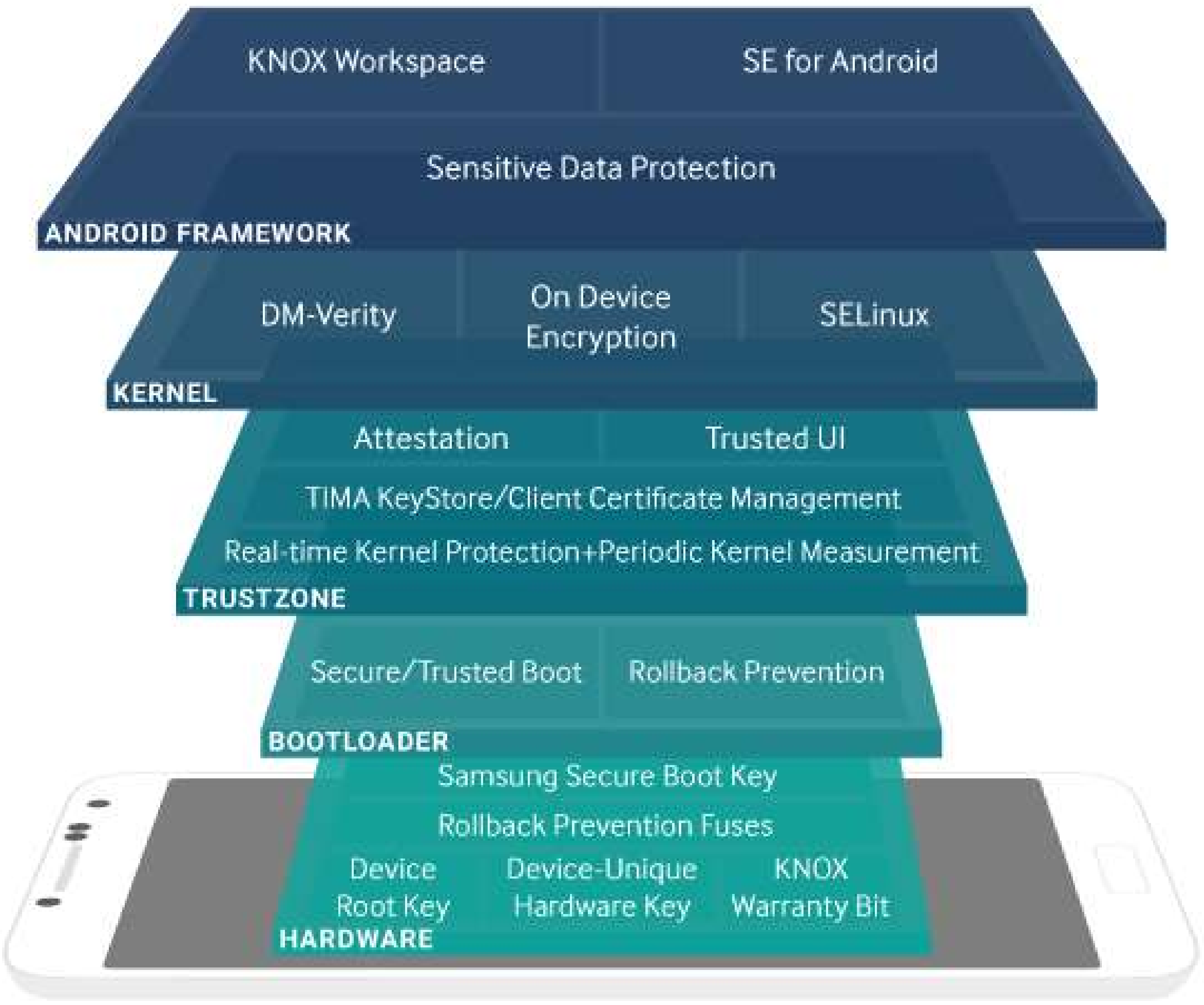

Figure 1 – Samsung Knox Security Solution

Table 1 summarizes the security challenges to an enterprise's Android adoption utilizing a Knox-specific solution.

Knox Strategy	Problem(s) Solved	Solution Technologies
Build a Hardware-Rooted Trusted Environment	Lack of trust in Android security	Hardware Root of Trust Samsung Secure Boot Key, Rollback Prevention Fuses, Knox Warranty Bit, *Device Root Key* (DRK) Build Trust Trusted Boot using *TrustZone-Based Integrity Measurement Architecture* (TIMA), Rollback Prevention Maintain Trust *Real-Time Kernel Protection* (RKP), *Periodic Kernel Measurement* (PKM), DM-Verity Prove Trust TIMA Attestation
Make Trusted Environment Enterprise-Ready	Mixing enterprise and user apps on one device	Knox Workspace, Security Enhancements for Android
	Device theft exposing enterprise data	Knox Workspace Encryption, *Sensitive Data Protection* (SDP), *On-Device Encryption* (ODE)
	Difficulty of securely implementing custom enterprise applications	*TIMA KeyStore, Client Certificate Manager* (CCM), *SE for Android Management Service* (SEAMS)
	Lack of enterprise manageability and utilities	*Enterprise Mobility Management* (EMM), *Virtual Private Network* (VPN), Active Directory Integration

Table 1 - Knox Solution Technologies

Problem: Lack of trust in Android security

The Android OS was originally designed for end users, not enterprises. The original Android approach to security was to simply isolate apps from interacting with one another. However, this design does not necessarily translate into enterprise confidence. For example, how can enterprises be sure security measures are enabled when users can root their device and intentionally exploit privileged software and circumvent vendor provided security measures?

Solution: Base security in a hardware-rooted trusted environment

Knox protects enterprise data by building a hardware-rooted trusted environment. A trusted environment ensures enterprise-critical operations can only occur when the device is in an allowed state. For software components such as the kernel and TrustZone apps, the allowed state is the required cryptographic signature of each piece of software. A trusted environment is hardware-rooted if both the cryptographic keys and code used to compute these signatures are tied back to unmodifiable values stored in hardware. Knox facilitates a hardware-rooted and trusted environment by:

1. Enforcing only approved versions of system-critical software be loaded
2. Ensuring system-critical software is not modified once loaded
3. Proving only approved system-critical software is loaded and run on a particular device when requested by the enterprise

Figure 2 on the next page shows how Knox builds, maintains, and attests its trusted environment.

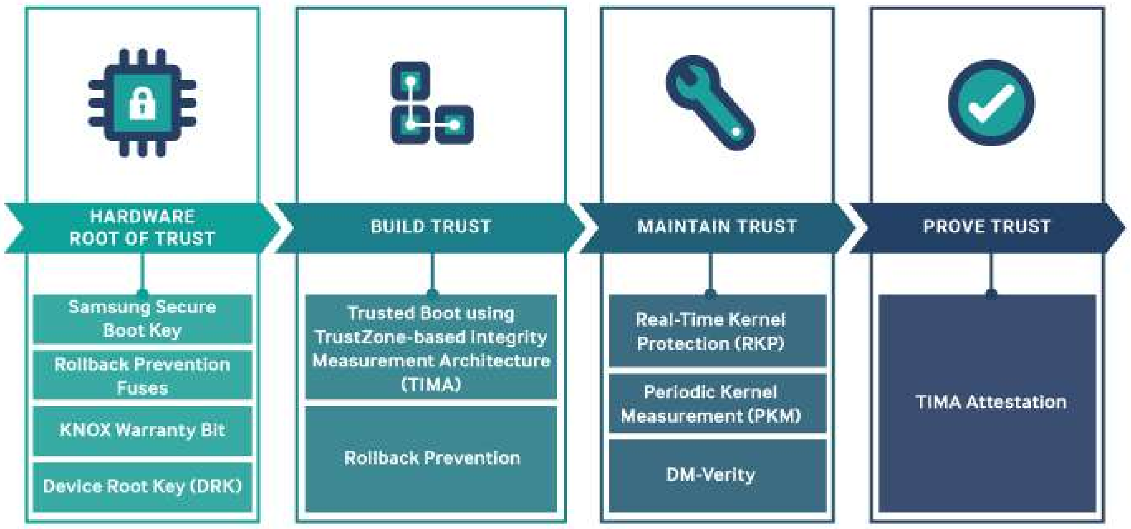
HARDWARE ROOT OF TRUST
Samsung Secure Boot Key
Rollback Prevention Fuses
KNOX Warranty Bit
Device Root Key (DRK)
BUILD TRUST
Trusted Boot using TrustZone-based Integrity Measurement Architecture (TIMA)
Rollback Prevention
MAINTAIN TRUST
Real-Time Kernel Protection (RKP)
Periodic Kernel Measurement (PKM)
DM-Verity
PROVE TRUST
TIMA Attestation

Figure 2 - Knox Builds, Maintains and Attests Its Trusted Environment

Building trust with secure and trusted boot

User applications are not immediately available when a device is powered on. Instead, a succession of software components initiate, with each component starting the next in the chain. Once initially powered, the hardware starts a bootloader running the operating system kernel. The kernel is a highly privileged component that starts applications and accesses storage and network devices directly.

Knox devices use *Secure Boot* to establish a successive component boot and signature recognition chain to verify the integrity of the each component. The device boot process halts if the signature verification process fails.

Secure Boot is limited, since it cannot distinguish between different approved versions, for example, a bootloader with a known vulnerability and a later patched version, as both versions have valid signatures. To address this limitation, Knox adopts Trusted Boot in addition to Secure Boot. With Trusted Boot, each software component in the chain measures and securely stores the cryptographic hash of the next component in TrustZone Secure World memory before loading it.

Storing and archiving measurement data enables third-parties to utilize *attestation* to identify the exact software version currently running on the device. This method ensures only the latest patched software versions are utilized to ensure patched software is not downgraded to an insecure version.

If signature verification fails and tampering is detected, Knox either blows a one-time fuse, called the Knox *warranty fuse*, or prevents further booting, depending on the configuration.

Both Secure Boot and Trusted Boot have their trust rooted in hardware. The first piece of software loaded is the primary bootloader, which resides in hardware-protected *Read-Only Memory* (ROM). The Samsung Secure Boot Key is the cryptographic key used to verify signatures, and is also stored in the device's hardware fuses.

Maintaining trust with runtime protection

Only Knox approved system software versions (such as the TrustZone OS) load at the end of a successful secure boot. Once loaded, they can then be modified. For example, users may either intentionally or unintentionally run code that exploits a flaw to maliciously modify the kernel, thus rendering it compromised. Knox detects kernel compromises quickly using *Real-Time Kernel Protection* (RKP) to actively prevent kernel code modification, and *Periodic Kernel Measurements* (PKM) that periodically check kernel code integrity. RKP checks occur in an isolated environment inaccessible to the kernel, so potential kernel exploitation cannot be extended to compromise RKP. Depending on the device model, this isolated environment can be either the TrustZone Secure World or ARM virtualization extensions. The ARM architecture virtualization extensions enable the implementation of an isolated virtual machine *hypervisor* to securely isolate RKP from the Android OS kernel. Both environments are hardware-protected and isolated from the Normal World. PKM checks occur within the TrustZone's Secure World.

The kernel is not the only attractive target for malware and malicious users. There are large numbers of other code objects and configurations that can be used by malware to become persistent, and restart each time the device restarts. Knox prevents such modifications by integrating Google's DM-Verity, a kernel module that verifies the integrity of applications and data stored on the critical system partition. If a malicious process or system partition modification is detected, DM-Verity flags the modification the next time the data is read, and blocks any attempt to access the modified data.

Proving trust

Consider an EMM server that wishes to interact with a mobile device. The EMM should not simply assume a device is not compromised. Instead, a Knox-enabled device provides the EMM with an *attestation*, a cryptographically verifiable collection of device state measurements. This attesation includes bootloader hashes, kernel, TrustZone, and logs from runtime protection mechanisms, among others. The EMM can then approve or reject the items on the list. Attestation is signed using a key derived from a hardware protected *Device Root Key* (DRK). A trusted environment is proved to a third party when the device hardware is validated as not tampered or exploited, and the ARM TrustZone Secure World software works properly. Knox solves the security threats described in subsequent sections of this whitepaper via technologies Samsung built on top of this trusted environment. Without the trusted environment in place, all remaining security measures are ineffective, as there is no guarantee they were even loaded or will run as expected.

Problem: Mixing enterprise data and user apps on one device

One of the major BYOD and COPE challenges facing enterprises is the convergence and interaction of sensitive enterprise apps and data with potentially malicious user-installed apps. Android provides apps with many ways to interact with one another. Apps may share databases containing photos or contact information, and perform actions on each other’s behalf, such as opening a link in an SMS text message. This interoperability has greatly benefited the mobile ecosystem, resulting in an explosion of useful applications. However, shared data from sensitive enterprise emails and documents can easily be leaked to untrusted apps claiming to provide a parallel task. Enterprises must guarantee their data is safe, even with hundreds or thousands of employees downloading untrustworthy third-party apps.

Solution: Protect enterprise apps and data in a secure Workspace

Samsung's strong isolation utilizes *Mandatory Access Controls* (MAC) to secure enterprise and user applications on the same device. With MAC, access to resources is restricted and can only be modified by the device vendor, in this case, Samsung.

Samsung adapted the *Security Enhancements for Linux* (SELinux) MAC system for Knox to provides a rich policy language for describing fine-grained access to resources by programs. Samsung extends SELinux into *Security Enhancements for Android* (SE for Android). SE for Android provides additional mediation locations in Knox's Android middleware, along with additional policy language. Knox's pioneering of SE for Android has led to its adoption into the *Android Open Source Project* (AOSP). The Knox *Workspace* is built on top of SE for Android to define a protected environment for apps and data. The workspace environment includes the home screen, launcher, applications, and widgets. The workspace functions alongside the user's environment, but it is protected from interference from user-installed applications. Data created by containerized applications are kept on a protected partition. Tampering detected during Trusted Boot execution renders the workspace and its data inaccessible.

Problem: Device theft

Smartphone theft is one of the most serious threats to confidential enterprise data.

A 2016 Consumer Reports study noted, “Smart phone thefts rose to 3.1 million last year,” and the report estimated that only 7% of smartphone users enable encryption for data at rest. Furthermore, only 36% enable a screen lock, and 34% take no security precautions at all. [5] Also, a recent FCC study estimates nearly 10% of all thefts and robberies in the US in 2013 were related to mobile device theft or compromised user data.[4] Consequently, a secure mobile OS must protect data without a user having to periodically invoke periodic measures.

Solution: Protect enterprise data-at-rest by default

Samsung Knox doesn't depend on device users to secure their own BYOD or COPE devices. Knox defines two data classes – *protected* and *sensitive*. All data written by apps in the secure workspace is considered Knox protected. Data is encrypted on disk when the device is powered off. In addition, the decryption key for protected data is tied to the device hardware. This makes protected data recoverable only on the same device. Access controls are used to prevent applications outside the Knox Workspace from attempting to access protected data.

Even stronger protection is applied to sensitive data. Sensitive data remains encrypted as long as the workspace is locked, even if the device is powered on. When the user unlocks their Knox Workspace using their password, *Sensitive Data Protection* (SDP) allows sensitive data to be decrypted. SDP keys are cleared when the user re-locks the workspace. The SDP data decryption key is tied to both device hardware and user input. Therefore, the data is recoverable only on the same device and with user input.

SDP can be used in one of two ways. First, all emails received are considered sensitive, and are immediately protected by SDP encryption. Emails received when the workspace is locked, are immediately encrypted, and can only be decrypted the next time the workspace is unlocked. The second way to use SDP is through the Knox *Chamber*. The Chamber is a designated directory on the mobile device file system. Any data placed into the Chamber is automatically marked as sensitive by Knox and protected by SDP.

Problem: Difficulty of securely implementing custom enterprise applications

Properly implementing cryptography, authentication, and secure storage services proposes unique challenges. Vulnerabilities exist in cryptographic libraries and applications that leak keys. Once a key is leaked, data previously encrypted with that key becomes vulnerable. Keeping secret keys secret is a problem for a number of reasons. First, many applications make multiple copies of keys in their internal logic, which they do not properly track and delete, thus increasing the risk of key leakage. Aggravating the problem, implementation flaws can expose secret keys directly to the network. In spite of the risks associated with implementing cryptographic services in applications, many enterprise apps require them.

Solution: Provide Knox security services to enterprise applications

Knox enables enterprise developers to build custom applications on top of its proven hardware-rooted trusted environment. Knox exposes APIs for key management, and FIPS 140-2 compliant cryptographic algorithms. The Trusted Boot-based TIMA KeyStore provides applications one type of secure key storage. Recall Trusted Boot only allows sensitive operations to occur if approved versions of all security-critical system software are loaded. The TIMA KeyStore stores all application keys in the TrustZone Secure World storage. From there, the keys can only be accessed if Trusted Boot is successful. Thus, an application's keys are safe, even in the event a user or malicious application tampers with critical system software.

Client Certificate Management (CCM) is similar to TIMA KeyStore as a complementary service for generating, storing and using asymmetric key pairs and certificates in TrustZone Secure World storage. The CCM API equips applications with PKCS#11 compliant token management, and public key algorithms for signatures and encryption.

Problem: Lack of enterprise manageability and utilities

Knox affords IT admins the functionality and manageability required for enterprise optimization. First, to use the secure environment effectively, enterprises need utilities such as a VPN and Microsoft Exchange integration. Second, enterprises need control of their devices and utilities to configure the workspace to meet their security needs.

Solution: Provide extensive manageability and utilities

Samsung Knox provides enterprises the controls to configure their workspace using an extensive set of more than 1500 APIs.

Samsung Knox utilizes per-application VPN controls, and smartcard framework integration with Microsoft Active Directory. Additionally, *Knox Mobile Enrollment* (KME) and *Knox Configure* (KC) are available to IT admins to securely enroll devices in bulk to exponentially reduce deployment times.

EMMs can prove their devices are running a trusted environment using TIMA attestations. TIMA attestation data contains boot component cryptographic hashes and security information. Data is signed using a key derived from the DRK, which proves that the attestation data originated from the TrustZone Secure World.

Samsung Knox has a dedicated team to help determine an enterprises' deployment needs and prepare custom Knox flavors if you enterprise requires customizations beyond what is offered by your EMM.

Samsung Knox has obtained a number of certifications that may be helpful if your enterprise requires compliance with specific security policies:

Certifications

FIPS 140-2 Certification	Issued by the *National Institute of Standards and Technology* (NIST), the *Federal Information Processing Standard* (FIPS) is a US security standard that helps ensure companies that collect, store, transfer, share, and disseminate *sensitive but unclassified* (SBU) information and *controlled unclassified information* (CUI) can make informed decisions when choosing devices for their workplace. Samsung Knox strictly adheres to FIPS 140-2 Level 1 certification for both *data-at-rest* (DAR) and *data-in-transit* (DIT).
DISA Approved STIG	The US *Defense Information Systems Agency* (DISA) publishes *Security Technical Implementation Guides* (STIGs) which document security policies, requirements, and criteria for compliance with DoD policy. DISA approved the STIG for Samsung Knox 2.x.
DISA Approved Product List	DISA has approved select Knox-enabled devices to the US DoD *Approved Products List* (APL). Note: Select Samsung Knox-enabled devices and tablets are certified under the *National Information Assurance Partnership* (NIAP) *Common Criteria* (CC) *Mobile Device Fundamental Protection Profile* (MDFPP).
Common Criteria Certification	The Common Criteria for Information Technology Security Evaluation, commonly referred to as Common Criteria, is an internationally-recognized standard for defining security objectives of information technology products and evaluating vendor compliance. A number of Governments use Common Criteria as the basis for their own certification schemes. Select Samsung Galaxy Knox-enabled devices received Common Criteria (CC) certification. The current CC certification targets the new *Mobile Device Fundamentals Protection Profile* (MDFPP) of the *National Information Assurance Partnership* (NIAP), which addresses the security requirements of mobile devices. Samsung Knox is approved by the US government as the first NIAP-validated consumer mobile devices to support its full range of classified information.

Certifications

CSfC	An ever increasing number of Samsung devices have been listed in the NSA/CSS's *Commercial Solutions for Classified Program* (CSfC) for approved security components.
ANSSI	Samsung Knox has obtained first-level security *Certification Sécuritaire de Premier Niveau* (CSPN) from the *Agence nationale de la sécurité des systèmes d'information* (ANSSI). The CSPN methodology and criteria is defined by ANSSI with evaluations run by ANSSI accredited testing labs.
ISCCC	Samsung Knox received the security solution certificate from the China *Information Security Certification Center* (ISCCC). Samsung worked closely with ISCCC to develop the certification process, including device requirements and security standards. By securing the critical ISCCC certification, Samsung has a stronger foothold to garner mobile device contracts with China's regulated industries, including government authorities, ministries, and finance.
CESG Approved	The *Communications and Electronic Security Group* (CESG) approved Knox-enabled Android devices for United Kingdom government use.
FICORA	Samsung Knox devices fulfill national security requirements as defined by the Finnish National Security Auditing Criteria (KATAKRI II).
ASD	Australian Signals Directorate is approved for ASD UNCLASSIFIED via MDFPP recognition.

NOTE: For the most recent Samsung Knox certifications, go to:
https://www.samsungknox.com/en/security-certifications

Section 4: Technology in depth

The Knox design philosophy consists of the following two steps:

1. Building a hardware-rooted and highly trusted environment.
2. Making the trusted environment enterprise ready.

Part 1. Building a hardware-rooted trusted environment

Knox builds a trusted environment in four ways. Knox first builds a hardware root of trust which other components rely. Second, Knox establishes trust during boot time. Third, Knox maintains trust while the device is in use. Finally, Knox proves its trustworthiness to remote parties, such as an enterprise management system.

Figure 3 displays an overview of the Knox architecture.

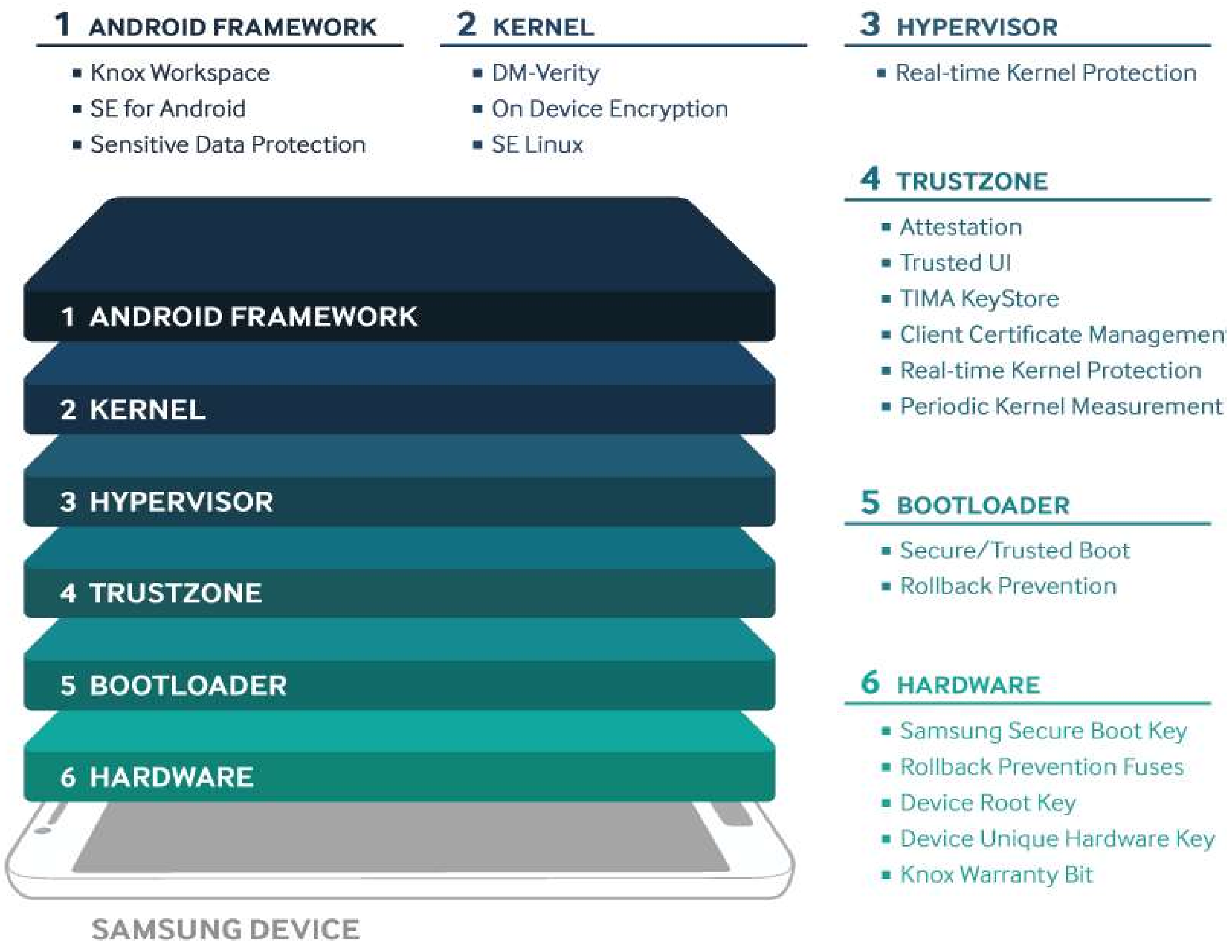

Figure 3 - Knox Architecture Overview

Hardware Roots of Trust

This section how Samsung Knox devices establish their trusted hardware environment.

Device-Unique Hardware Key (DUHK) Samsung incorporates the DUHK, a device-unique symmetric key, in the device hardware during manufacturing. The DUHK binds data to a particular device and is only accessible to a hardware cryptography module and not directly exposed to any device software. However, software can request that the DUHK encrypts and decrypts data. Data encrypted by the DUHK is bound to the device, since it cannot be decrypted on any other device.

Samsung Secure Boot Key (SSBK) The SSBK is an asymmetric key pair used to sign Samsung-approved boot executables. The private part the SSBK is used by Samsung to sign secondary and application bootloaders. The public part of the SSBK is stored in hardware one-time programmable fuses at manufacture time in the Samsung factory. The Secure Boot process uses this public key to verify whether each boot component it loads is approved.

Rollback Prevention Fuses (RP Fuses) RP fuses are hardware fuses that encode the minimum acceptable version of Samsung-approved bootloaders. Old software may contain known vulnerabilities that can be exploited. Rollback prevention prevents approved, but out-of-date versions of bootloaders from being loaded. The RP fuse version number is set when system software is initially installed and as Knox updates occur. RP fuses are programmable just once per Knox update.

Knox Warranty Fuse Knox utilizes a one-time programmable fuse that signifies whether the device has ever been booted into an unapproved state. If the Trusted Boot process detects non-approved components are used, or if certain critical security features such as SELinux are disabled, it sets the fuse. Thereafter, the device can never run Samsung Knox, access to DUHK and DRK in the TrustZone is revoked, and the enterprise's data on the device cannot be recovered.

ARM TrustZone Secure World The Secure World is a hardware-isolated environment in which highly sensitive software executes. The ARM TrustZone hardware enforces memory and devices marked secure can only be accessed in the Secure World. Most of the system as we know it, including the kernel and middleware, as well as all apps, execute in the Normal World and can never access the data used by Secure World software. The Secure World software, on the other hand, is more privileged, and can access both Secure and Normal World resources.

Bootloader ROM The *primary bootloader* (PBL) is the first piece of code to run during the boot process. The PBL is trusted to measure and verify the boot chain (see the sections on Secure Boot and TIMA Trusted Boot). To prevent tampering, the PBL is kept in secure hardware *Read Only Memory* (ROM). The device hardware loads and runs the PBL from ROM at boot, and the PBL starts the Secure and Trusted Boot processes.

Device Root Key (DRK) The DRK is a device-unique asymmetric key pair that is signed by Samsung's root key through an X.509 certificate. This certificate proves that the DRK was produced by Samsung. The DRK is generated at manufacture time in the Samsung factory and is stored on the device encrypted by the DUHK, thus binding it to the device. The DRK is only accessible from within the TrustZone Secure World.

Because the DRK is device-unique, it can be used to tie data to a device through cryptographic signatures. The DRK is not used directly to sign data; instead, signing keys are derived from the DRK. The TIMA attestation data, proving the device is in a trusted state, is signed using the Attestation Key, which is itself signed by the DRK. The DRK signature proves attestation data originated from the TrustZone Secure World on a Samsung device. Note that while the DRK is not stored directly in hardware, it is an important part of the root of trust, as it derives other signing keys, and is protected by both the DUHK and TrustZone Secure World.

Establishing trust

Android begins the startup process with the primary bootloader, which is loaded from ROM. This code performs basic system initialization and then loads another bootloader, called a secondary bootloader, from the file system into RAM and executes it. Multiple secondary bootloaders may be present, each for a specific task. The boot process is sequential, with each secondary bootloader completing its task and executing the next bootloader in the sequence, finally loading the application bootloader known as *aboot*, which loads the Android operating system. This sequence is called the boot chain.

Secure Boot

With a Secure Boot, each component in the boot chain verifies the integrity of the subsequent component against a signature before executing it. Knox stops the boot process if verification fails. Boot component signatures generate at build time using the *Samsung Secure Boot Key* (SSBK). The public part of the SSBK is stored in hardware fuses during manufacture. The first component in the chain, the primary bootloader, is stored in immutable ROM and is trusted to verify the secondary bootloader. Thus, the Secure Boot chain can only be compromised by hardware tampering. Later boot components, such as the kernel, are signed by another Secure Boot Key programmed into the previous boot component.

TrustZone-based Integrity Measurement Architecture

Secure Boot prevents a device from starting if unapproved boot components are detected. However, if the device does start, Secure Boot cannot inform a third party about what approved boot components have been loaded and run. For example, it cannot distinguish between a boot component with a known vulnerability versus a later patched version, since both versions have valid signatures. In addition, some carriers may decide to allow custom OS kernels to run on their devices. On these devices, Secure Boot cannot prevent unapproved kernels from running. This restriction poses a threat to enterprise applications and data. To remedy this Secure Boot limitation, Knox contains the *TrustZone-based Integrity Measurement Architecture* (TIMA). TIMA utilizes two features: Trusted Boot and Attestation.

TIMA Trusted Boot

In Trusted Boot, each boot component in the boot chain measures a subsequent component and stores the measurement before executing it. The Trusted Boot process flow is displayed in Figure 4, using a SHA256 cryptographic hash of the boot component. These hashes are securely stored in TrustZone-protected memory. The hash sets consist of one or more secondary bootloaders, the TrustZone Secure World operating system, the application bootloader, and the Normal World kernel. Depending on the processor make and model, additional firmware image hashes, such as the modem are included. These hashes validate device integrity to a remote server using TIMA Attestation.

Low level components tightly tied to the device hardware, such as bootloaders, should never be replaced. Any attempt to replace a low level component results in prompt informing the user to take the device to a service center for administration.

If the kernel has been modified, Trusted Boot sets the Knox warranty violation fuse. The one-time programmable memory fuse indicates the device has been tampered and cannot invoke certain Knox features thereafter. Even if the boot code is restored to its original factory state, tampering evidence remains and is reflected in the attestation results. Some device models will opt to never set the warranty violation fuse, instead always requesting the user service the device.

As bootloaders execute and takes measurements, those measurements are stored in TrustZone secure memory for future inspection using attestation.

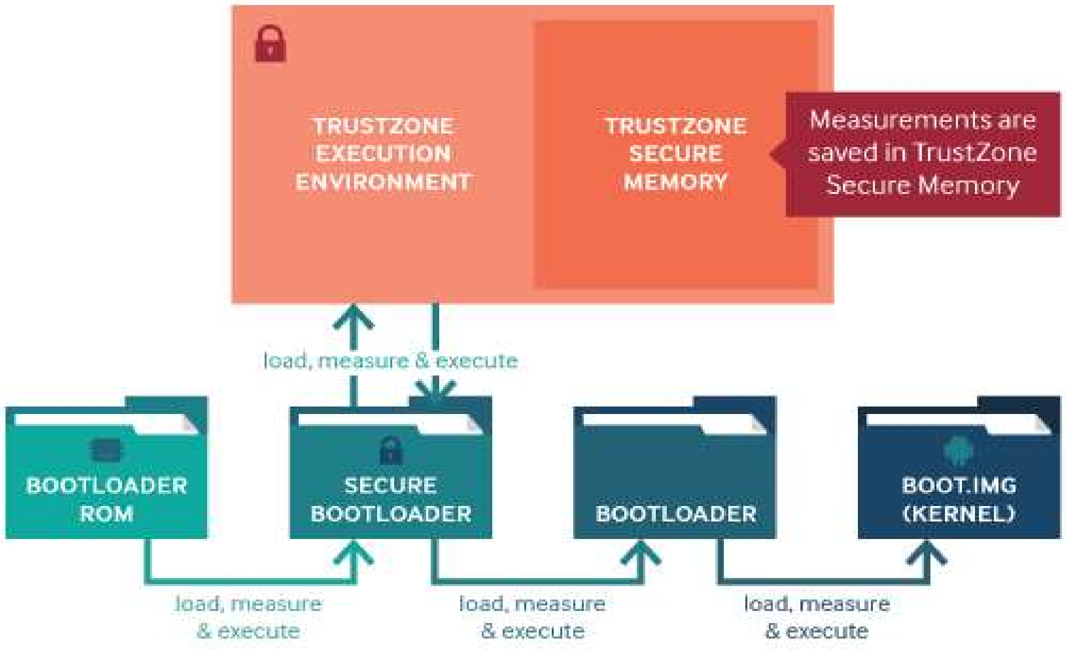

Figure 4 - The Trusted Boot Process

Rollback Prevention (RP)

Rollback Prevention blocks the device from loading or flashing an approved but old version of boot components. Old versions of software may contain known vulnerabilities that attackers can exploit. Rollback prevention checks the version of the bootloader and kernel during both boot and updates, and blocks these processes from continuing if versions are unacceptably old. The lowest acceptable version of the bootloader is stored in secure hardware fuses when the device is flashed, and the lowest acceptable version of the kernel is stored in the bootloader itself. Whenever a vendor-applied update occurs, the lowest acceptable version can be incremented in the fuses. Because this value is kept in fuses, it cannot be decremented even through physical tampering.

Maintaining trust

The following describe how Knox maintains hardware trust by protecting kernel data.

Periodic Kernel Measurement (PKM)

TIMA PKM conducts periodic monitoring of the kernel to detect if legitimate kernel code and data have been exposed to malicious software. In addition, TIMA also monitors key SE for Android data structures in OS kernel memory to detect malicious attacks could corrupt and potentially disable SE for Android.

Real-time Kernel Protection

Kernel security is essential to the Knox system. An attack that compromises the kernel has the ability to arbitrarily access system sensitive data, hide malicious activities, escalate the privilege of malicious user processes, change the system behavior, or simply take control of the system. As previously mentioned, Trusted Boot measurements determine which kernel was loaded and run when the device was started. However, this protection does not guarantee the integrity of the kernel after the system begins to interact with potential attackers. Clever attackers can often exploit an already booted and running kernel. In such cases, it is important to continuously monitor the kernel during system runtime to detect and prevent modifications to the kernel code or critical data structures.

Intuitively, the kernel protection mechanism cannot itself exist completely in the kernel, or it could be circumvented by an attacker. Therefore, Samsung Knox introduces *Real-time Kernel Protection* (RKP), a unique solution that provides the required protection using a security monitor located within an isolated execution environment. Depending on the device model, this isolated execution environment is either the Secure World of ARM TrustZone or a thin hypervisor that is protected by the hardware virtualization extensions. RKP's *Trusted Computing Base* (TCB) is part of this isolated environment and thus is secure from attacks that may potentially compromise the kernel.

Running in an isolated environment can hinder a security mechanism's ability to closely monitor events occurring inside the target kernel. To remedy this problem, RKP uses special techniques to control Normal World memory management and intercept critical events and inspect their impact before allowing them to execute. Therefore, RKP complements TIMA-PKM's periodic kernel integrity checks and their limited effectiveness against attacks and hide their traces between checks.

RKP achieves three important security initiatives:

- First, RKP prevents running unauthorized code on the system, which is accomplished by preventing the modification of kernel code, the injection of unauthorized code into the kernel, or the execution of the user space code in the privileged mode.
- Second, RKP prevents kernel data from being accessed directly by user processes. This includes the double-mapping of physical memory containing critical kernel key data into user space virtual memory. This is an important step to prevent kernel exploits that attempt to map malicious processes to kernel data regions where such regions could be modified by an attacker by an attacker.
- Third, RKP monitors some critical kernel data structures to verify that they are not exploited. In particular, RKP protects the data that defines the credentials assigned to running user processes to prevent attackers from modifying this data.

Additional Knox protection features are continually under development.

Architecture overview

Figure 5 displays the RKP architecture hosted in an isolated execution environment that's protected even if Android's Linux kernel is compromised. The kernel pushes a request RKP to perform two operations on its behalf: (1) emulate control instructions that change the system state, and (2) update the Normal World memory translation table.

System control instructions allow the Normal World to control the security critical system state, such as defining the location of memory translation tables and exception handlers. These instructions can only be executed by privileged code, such as the kernel code. RKP works with the kernel so certain system control instructions are removed from its executable memory (the only memory-executing privileged instructions in the Normal World). Consequently, the only way to execute these instructions is by emulating them in the Secure World. Samsung calls this operation *Control Instruction Emulation*. On models using virtualization extensions, intercepting system control instructions is accomplished using hardware virtualization extensions.

Memory translation tables define the virtual-to-physical address mapping and access permissions of virtual memory. If the kernel attempts to change the current memory layout by modifying the translation tables, then RKP inspects the changes to confirm they do not impact system security. RKP ensures translation tables cannot be modified by the Normal World by making them read-only to the Normal World kernel. Therefore, the only way the kernel can update translation tables is to request the updates from RKP. As a result, RKP guarantees this security design is non-bypassable.

Kernel code protection

Kernel code protection is RKP's central feature and benefit. An attacker bypassing Linux kernel defenses is not allowed to modify the kernel executable code, significantly reducing the vulnerability of kernel attacks to the whole system. RKP examines memory translation table modifications to enforce rules so the kernel is not writable by code in the Normal World.

For detailed information on RKP and the TrustZone-based implementation of RKP, go to the ACM Digital Library website: http://dl.acm.org/citation.cfm?id=2660267.2660350&coll=DL&dl=GUIDE&CFID=629439201&CFTOKEN=91386218.

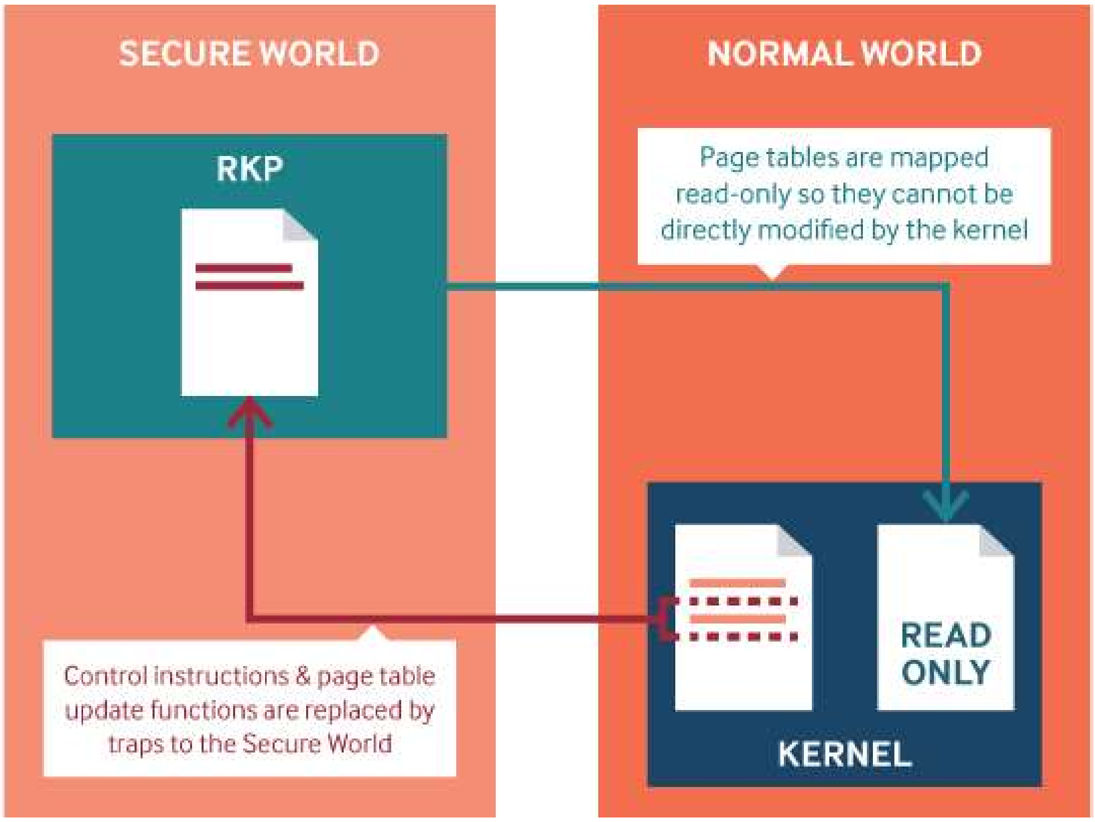
SECURE WORLD
NORMAL WORLD
RKP
Page tables are mapped read-only so they cannot be directly modified by the kernel
READ ONLY
KERNEL
Control instructions & page table update functions are replaced by traps to the Secure World

Figure 5 - RKP Architecture

RKP interoperability rules guarantee the RKP monitoring function cannot be bypassed, even when an attacker violates Normal World kernel protections. As a result, the kernel cannot modify its own code, even if compromised.

RKP rules include:

- Kernel code pages are never mapped writable un der any condition
- Kernel data pages are never mapped as executable
- Memory translation tables are mapped read-only to the Normal World
- Double mapping the kernel code or the memory translation table is not allowed. Double mapping occurs when the same physical memory is mapped to multiple virtual memory addresses, which could permit two different parts of the system to access the same memory with different permissions.
- All mapped memory regions should have the *Privileged eXecute Never* (PXN) permission, with the exception of the OS kernel.

The first two rules guarantee the initial kernel image measured by Trusted Boot, cannot be directly modified by a potential attacker unless it changes the system's memory mapping by modifying the memory translation tables. These rules remain true even if the attacker takes control of the kernel itself. The rest of the rules guarantee memory translation tables cannot be modified by the kernel, unless it sends a request to RKP. If a request is sent, RKP verifies the translation table modification does not violate the above rules. The kernel is not modified without RKP's knowledge if combining these two sets of rules together.

The kernel code protections discussed in this section assume the system memory management state has not been modified. Modifying the memory management system state (changing the effective memory translation tables' base address, or disabling virtual memory protection completely) could allow an attacker to bypass RKP monitoring. Therefore, RKP uses *Control Instruction Emulation* to inspect events to guarantee they do not tamper with its monitoring.

Knox traps system control instructions into RKP using hardware controls. In models using a TrustZone-based solution, this feature is constrained by TrustZone being incapable of trapping changes to the Normal World state. Therefore, RKP instructs the kernel to remove each system control instruction. Since these instructions can only run from privileged code, and RKP grants that privilege exclusively to the measured and protected kernel code, it is impossible for the Normal World to run these instructions without trapping to RKP. In turn, RKP validates the values written to the system control instructions to guarantee they do not invalidate its kernel code protection assumptions.

Preventing double mapping of kernel data

Kernel data structures are critical to device security. Maliciously modifying kernel data can lead to a wide range of damage. RKP uses two methods to protect the kernel codebase and prevent return-to-user attacks that exploit the operating system kernel and enable users to hijack privileged execution paths with escalated privileges. The first is through double mapping the memory hosting kernel data into the address space of the malicious process. The second is to alter the kernel control flow so it maliciously modifies its own data (such as using pointer manipulation or pointer overflow). The first attack class, double mapping kernel data to malicious user processes, is a real threat to the kernel. For instance, a real-world Android exploit used an integer overflow to trick the kernel into mapping a huge amount of the physical memory into the address space of the attacking process.
To prevent the malicious double mapping of kernel data, RKP ensures physical memory pages hosting this data are not mapped to user space processes. They can only be mapped as privileged pages and cannot be

accessed by the user space. RKP enforces this rule using its control of Normal World memory translation. RKP rejects any page table modification mapping kernel data to the user space. To handle a related problem, RKP ensures no executable kernel pages are ever double-mapped as writeable, and vice versa.

RKP relies on the target kernel to inform it about the location of critical data. RKP embeds hooks in the kernel code so it is informed whenever a new memory area is allocated to the kernel. It then prevents this memory from being double mapped to writable memory anywhere else on the device.

This protection is effective against attacks using double mapping to exploit kernel data. Although RKP relies on the kernel to inform it about allocated data memory areas, this dependency does not weaken the protection. The kernel is assumed secure when it sends the information to RKP, since the data exchange occurs before the data pages are allocated. Afterwards, RKP prevents the data from being modified, except by the kernel itself.

Protecting the kernel data that defines user process credentials

The last class of attacks threatening kernel security is the alteration of the kernel control flow so it maliciously modifies its own data. These attacks may include pointer manipulation, pointer overflow, or return-oriented attacks.

Although RKP cannot fully protect against user process credential kernel attacks, it implements a novel technique to mitigate their effect by protecting selective kernel data structures critical to the system security. The data structure of choice is the *process credentials* data structure, which defines the privilege level of the user processes running inside the device. User processes represent different running applications, such as user apps. In Linux, there is an instance of the credentials structure associated with each running process. These credentials are frequently the target of rooting attacks, since a normal process can elevate its privilege through the exploitation of the credentials.

RKP uses a three-step solution to protect the credential structure from malicious modifications. First, RKP makes each instance of the credential data structure read-only by controlling the memory translation tables. Second, RKP instructs the kernel so writes to the credential structures are routed through RKP. The kernel is now unable to write to this data from within the Normal World. Before writing to the credential data, RKP examines the values to be written to make sure they do not maliciously escalate the privileges of their corresponding user process. Determining if a user process is legitimately entitled to an escalated privilege, such as the administrative privilege, is accomplished by combining multiple techniques. For example, RKP prevents processes that start with regular user privilege from escalating their privilege after they start. Additionally, processes started by applications that interface with potential attackers, such as zygote and shell, are not allowed an escalated privilege. Finally, RKP adds a check to the kernel security hooks to verify a credential structure actually belongs to the read-only memory protected by RKP before it determines the privilege of the user process. Therefore, it is guaranteed that a potential attacker cannot forge a malicious instance of the credential structures that are not monitored and verified by RKP.

For detailed information on RKP and the TrustZone-based implementation of RKP, go to the ACM Digital Library website: http://dl.acm.org/citation.cfm?id=2660267.2660350&coll=DL&dl=GUIDE&CFID=629439201&CFTOKEN=91386218.

DM-Verity

Attackers may intend to expand their exploits beyond modifying bootloader or kernel images. There are other software binaries and configuration files in storage which provide malware persistence. Persistent malware is able to restart itself each time the system is rebooted. The malware restarts by modifying programs or configurations on the system partition that contain the system binaries, Android framework, and configuration files that were started during boot. Malware can survive system reboots once inserted in the boot path. Additional problems can arise from tampering with system data and configurations, such as the granting of excessive privileges to vulnerable applications.

To prevent unauthorized modification to the system partition, Knox integrates a customized DM-Verity implementation that's a Linux/Android kernel module that performs integrity checks on all data blocks contained in a block device (such as a partition).

With stock Android, DM-Verity uses a hash tree to conduct integrity checks of individual data blocks. The hash root tree is signed by an RSA key. Whenever a data block is read into memory, DM-Verity computes the hash of the block, and then uses it, along with the other hashes on the path to the root to compute the root hash. If this computed root hash matches the signed version, the block is considered good. Otherwise, unauthorized modification of the block is detected, and the access to the data block is restricted.

Knox's DM-Verity implementation differs from stock Android in supporting file-based *firmware over-the-air* (FOTA) software updates. The Knox approach is easier to support with existing infrastructure than the stock block-based approach.

Proving trust

TIMA Attestation

TIMA Attestation enables a device to convey state information to a remote server, such as an EMM server. The attestation message contains state measurements that can be evaluated by a server, which can then decide whether to trust the device or not. A typical message contains:

- Measurements collected by Trusted Boot to demonstrate only approved system software was loaded during boot
- Security violation logs from PKM and RKP since the last reboot
- Knox warranty violation fuse status
- Device-identifying information, such as the IMEI and Wi-Fi MAC address
- A locally-computed verdict whether the device is trustworthy

The full attestation message is computed in the ARM TrustZone Secure World, and is accurate even if the Normal World OS is compromised. The verdict is a central part of the attestation message. Only when both the measurements collected by Trusted Boot match known good values, and the warranty violation fuse is intact is the verdict set to Yes to indicate attestation has passed. Good measurements are retained in a file called tima_measurement_info, maintained in TrustZone secure storage. This file is generated at build time. To simplify remote server logic, they can directly use the verdict instead of verifying all the measurements themselves.

An attestation message cannot be forged, since it's signed using the TIMA Attestation Key (and traceable to Samsung's root key). Each Samsung device supporting TIMA attestation has a unique RSA key pair, called the *Device Root Key* (DRK). The DRK is generated during manufacture and is traceable to Samsung's root key using X.509 certificates (stored in TrustZone). The remote server can verify message integrity using Samsung's root key. The signature includes a server-generated cryptographic nonce (a random number used only once) to ensure an attacker cannot replay old valid attestation messages on an already compromised device.

To illustrate this capability, consider the EMM server example previously described earlier. Depending on the attestation data and verdict, any further action is determined by the enterprise EMM security policy. The security policy might choose to detach from the device, erase the contents of the secure workspace, ask for the device's location, or any of many other potential device recovery procedure.

Part 2. Making the trusted environment enterprise ready

The next several sections describe the technologies constructed in the trusted environment to provide Knox enterprise optimizations.

SE for Android

Samsung Knox adopts the *Security Enhancement for Android* (SE for Android), which adds *Mandatory Access Control* (MAC) to *Android. Discretionary Access Control* (DAC) mechanisms, such as Android permissions or Linux owner/group/world permissions, have few security benefits since the user or process generating data has the ability to change the data's access rules. A user can make bad decisions with the data, which may then be leaked publicly. MAC provides security experts with enforcement rules that can’t be maliciously or ignorantly overridden by device users or software developers. Since these rules are mandatory, and cannot be altered, they help prevent malicious code or untrusted users from accessing sensitive data or programs. MAC can lock down data a user may want to keep secret, and prevents developers from maliciously or accidentally compromising system components that protect our devices.

SE for Android provides two layers of MAC protection:

1. Kernel-level protection: Android inherits its SELinux MAC capability directly from Linux. SELinux provides MAC for kernel system calls. A SELinux policy can enforce which objects system calls can target. For example, specify only system-signed processes can read files in the data/security directory. This level of control is possible because access check hooks are inserted inside the kernel. These hooks query the security policy before each system call to determine if it's an allowed action. SELinux policies can prevent processes from reading or tampering with data, bypassing security mechanisms, or interfering with other processes. They also reduce the damage from malicious or flawed programs

2. Android middleware protection: There's many parts of the Android system that do not leverage system calls. For example, the Android Intents used to start apps. The layer above the kernel, but below user space applications, is called the Android middleware. Additional hooks have been added to key decision points to extend MAC control to the middleware. This is known as *Middleware MAC* (MMAC). MMAC can enforce security policies among inter-component communication for Android Apps.

SE for Android security objectives include strong data and application isolation, confining the permissions of system processes running as root, and protecting applications.

Scope of access control

Samsung's custom version of SE for Android provides the following unique features:

- MAC on APIs (control who can invoke your APIs)
- Knox Workspace isolation of personal & business data
- On-the-fly workspace creation for security cusomtizations
- Quick-response policy updates (no carrier-approved firmware updates required to plug vulnerabilities)
- Strong application isolation beyond Android's access control
- Extensible MAC for new Knox features

Samsung also built an innovative global policy validation system that can detect when prohibited actions are attempted. Early detection affords Samsung unique visibility into how its devices are used and provides an earlier window to mitigate new threats before they can be exploited. This policy validation system can also refine Samsung's ability to accurately grant only the permissions needed.

SE for Android policy

SE for Android includes a set of security policy configuration files designed to meet common, general-purpose security goals. Out of the box, Samsung Knox provides a policy designed to strengthen the core Android platform and exceed enterprise needs. Samsung Knox also provides a SE for *Android Manager Service* (SEAMS), with management APIs allowing enterprise IT admins to manage SE for Android. Management tasks include gathering access logs, resetting file security labels, mapping applications to different security domains, getting type context information, and obtaining status information about packages and workspaces.

Samsung Knox provides policies to enforce the isolation of application workspaces. For example, Samsung Knox contains new security domains and can now enforce *Multiple Category Securit*y (MCS) isolation. Categories isolate applications and data into security groupings, independent of what security domain they're assigned. Categories can then ensure personal and business applications with the same security domain have their access rights limited to just their own areas. New workspaces can also be created on the fly by simply applying a new security category to a group of apps.

Sensitive Data Protection

Knox enforces two protection classes for data generated within the workspace: protected data and sensitive data. All the data generated from within the Knox Workspace is considered protected. Protected data residing in storage is always encrypted, and protected against offline attacks. Additionally, access controls prevent applications outside the workspace from attempting to access protected data. The decryption key for protected data is stored encrypted by the device-unique hardware key (DUHK). Therefore, the key is only recoverable on the same device.

Sensitive data provides an even stronger security guarantee. Like protected data, sensitive data is always encrypted when on disk. Additionally, the data remains encrypted as long as the workspace is locked. The key used to decrypt sensitive data on disk is recoverable only if the user enters the workspace password, PIN, or pattern. Thus, if a device is stolen, the key cannot be extracted from the device. Like protected data, the stored key material is encrypted by the DUHK, binding it to the device.

The enforcement of the sensitive data guarantee is conducted using Knox *Sensitive Data Protection* (SDP). SDP creates a *Container Master Key* (CMK) that can only be decrypted with user input. If desired, an EMM can also be used to unlock the CMK, preventing a total data loss in the event of a forgotten workspace password. Once the workspace is locked, SDP clears the keys in memory after a configurable timeout interval (five seconds by default). In addition, SDP also flushes sensitive file data from the OS kernel's disk cache if the file is not in use by a workspace application.

Any sensitive data received if the workspace is locked is still protected by SDP using a public key algorithm where the private part of the key is maintained in an encrypted partition, and the public part encrypts the new sensitive data. Once the workspace is unlocked, the data is decrypted with the private key, and re-encrypted using the usual symmetric key, guarded by the CMK. Currently, email subjects, bodies, and attachments are marked sensitive. Additionally, the SDP Chamber provides a designated directory on the file system. Any data placed into the Chamber is automatically marked as sensitive and protected by SDP.

On-Device Encryption

In addition to Android's kernel-level device encryption, Knox ties the encryption key to a secret maintained in trusted hardware. This is only available if the enterprise IT admin activates encryption via the EMM. TrustZone-based AES 256 *on-device encryption* (ODE) also enables enterprises to ensure device data is protected in the unlikely event the operating system is compromised. While this feature is low overhead, providing system-wide encryption means less flexibility in supporting separate security levels for user and enterprise data, thus the inclusion of the finer-grained protected and sensitive data classes.

Trusted Boot Based KeyStore (TIMA KeyStore)

The TIMA KeyStore provides applications with services for generating and maintaining cryptographic keys. The TIMA KeyStore is only enabled if the Trusted Boot measurements match the known good values in the tima_measurement_info file, and if the Knox warranty fuse is not set. Consequently, cryptographic operations with keys in the KeyStore can only occur if the system was booted into an approved state. Keys stored in the TIMA KeyStore are further encrypted with the *device-unique hardware key* (DUHK), and can only be decrypted from within TrustZone Secure World on the same device. Cryptographic operations on the keys are performed within TrustZone Secure World.

The TIMA KeyStore has the same API as familiar Android KeyStore APIs. Therefore, the only modification necessary is to specify the TIMA KeyStore used to provide the service.

Trusted Boot Based Client Certificate Management (TIMA CCM)

TIMA CCM permits the storage and retrieval of digital certificates, as well as encryption, decryption, signing, and verification in a manner similar to Smartcard functions. The certificates and associated keys are Knox encrypted with a device-unique hardware key that can only be decrypted from code running within TrustZone.

TrustZone-based CCM also provides the ability to generate a *Certificate Signing Request* (CSR) and the associated public/private key pairs to obtain a digital certificate. A default certificate is provided for applications not requiring their own certificate.

Programming interfaces for certificate storage and management are provided in the Knox Premium SDK. Application developers are provided with industry standard PKCS #11 APIs for certificate management, and therefore interact with the CCM as if it were a virtual Smartcard. Like the TIMA KeyStore, TIMA CCM operations are permitted only if the device was booted into an approved state.

Trusted UI

Knox provides a Trusted UI for secure credential entry for enterprises using PIN-based authentication. The Trusted UI uses ARM TrustZone to create a dedicated path from the device's screen and keyboard to the Secure World. Credentials entered while this path exists are completely inaccessible to Normal World programs and untrusted peripherals. Once the credentials are held in the Secure World, they are passed back to the enterprise application that initiated the authentication request.

Data erase during factory reset

Samsung's device reset procedure restores device software to its original factory default settings. The reset is completed before changing device ownership or disposing the device. Securely removing existing user data so no data is recoverable after the reset is a critical.

Erasing data on flash storage requires extra care. Samsung devices store data in a type of flash storage called *embedded multimedia cards* (eMMC). eMMC firmware uses translation tables that map device-visible logical memory to flash physical memory to improve performance and card life. This means devices cannot reference physical flash memory directly, and thus cannot ensure data is erased without support from the eMMC itself.

Samsung devices use several features supported by Samsung-manufactured eMMC chips to ensure a data erase operation during factory reset. First, when the user initiates a factory reset, the reset code instructs the eMMC firmware to discard the entire physical memory range corresponding to the logical memory storing user data. Discarded user data thereafter returns zeros when accessed by the device OS. Second, the Samsung eMMC controller firmware code responsible for discarding the physical memory is itself protected against malicious updates.

Workspace data resides encrypted in flash memory, offering yet another layer of data protection. The cryptographic keys used to encrypt Knox Workspace data are themselves stored encrypted by the device-unique hardware key, accessible only by a separate secure processor.

Section 5: Enterprise readiness

Knox Workspace: Divide and conquer

Knox Workspace is a dual persona container designed to separate, isolate, encrypt, and protect enterprise data from attackers. This work/play environment ensures work and personal data is separated, and only the work container is managed by the enterprise. Personal information like photos and messages are not managed or controlled by the IT department.

The applications and data inside workspace are isolated from applications outside workspace. Consequently, applications outside the workspace cannot use Android inter-process communication or data-sharing with applications inside the workspace. For example, photos taken with the camera inside workspace are not viewable in the Gallery outside workspace. The same restriction applies to copying and pasting. When allowed by an IT policy, some application data, such as contacts and calendar data, can be shared across the workspace boundary. The end user can choose whether to share contacts and calendar notes between the workspace and personal space. However, an IT policy ultimately controls this option.

An enterprise can manage the workspace like any other IT asset using an EMM solution; this container management process is called *Mobile Container Management* (MCM). Samsung Knox supports many of the leading solutions on the market. MCM is affected by setting policies in the same fashion as traditional EMM policies. Samsung Knox Workspace includes a rich set of policies for authentication, data security, VPN, e-mail, application blacklisting, whitelisting and so on.

Upon creation, IT admins can choose the workspace UI style (folder or launcher style), and can prevent end users from making further style changes.

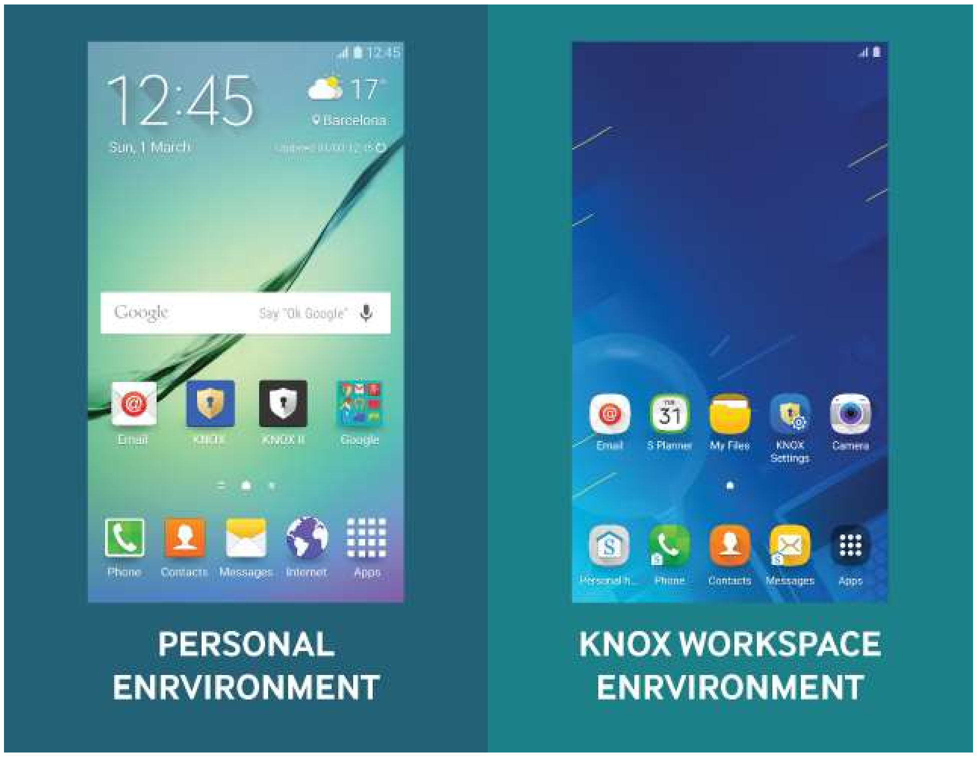

Figure 6 - User's personal environment running next to the workspace environment

Users can distinguish between the utilization and population of their personal environment applications versus workspace environment applications.

Knox Workspace also utilizes two-factor authentication. A user can set the workspace to accept a fingerprint or iris scan as the primary authenticator with a PIN, password or pattern as a second factor. The iris scan biometric authentication method is available on the Galaxy S8 platform and beyond.

The Knox platform also supports two workspaces when needed, thus meeting the needs of professionals using their own devices for corporate use and have multiple employers, such as doctors or consultants.

Other workspace features include optional Bluetooth® and *Near Field Communication* (NFC) inside the workspace itself. NFC enables a device to act as a SmartCard-based credential for physical device access and access to IT accounts. Bluetooth can be used to communicate with connected devices, and support Bluetooth profiles to enable additonal support, including printing, file sharing, and external card readers.

Applications inside the workspace can also connect with USB accessories, such as a USB printer. To propely secure the connection, IT admins must explicitly allow the USB between connectiuon between the container apps and external storage. The default for mass storage is set to OFF, and is controlled by an enterprise IT admin policy.

For Samsung Note users, a S-Pen Air Command is also supported in the workspace for writing memos, adding personal app shortcuts, screen captures, and writing notes on a screen capture (depending on the IT policy).

Knox caller ID for incoming calls, when in Personal mode, can also be configured by IT admins to display caller ID information derived from both personal contacts and Knox Workspace contacts.

Google Play for Work

IT admins can install Google Play for Work inside the Knox Workspace to silently install and uninstall apps and optionally blacklist or whitelist apps. Enterprise employees can also download IT admin approved apps in Knox Workspace. Google Play for Work can also be used outside workspace.

Additonallly, Google Voice apps inside the Knox Workspace enable users to utilize voice recognition in addition to the touchscreen keyboard.

Sensitive Data Protection (SDP) and Knox Chamber

SDP can be used in one of two ways. First, all emails received are considered sensitive, and are immediately protected by SDP encryption. Emails received when the workspace is locked, are immediately encrypted, and can only be decrypted the next time workspace is unlocked.

The second way to use SDP is through the Knox Chamber. The Chamber is a designated directory on the file system and a user-accessible folder inside the workspace. Any data placed into the Chamber is automatically marked as sensitive and protected by SDP.

Third-party application data can also be encrypted when a device is locked, then decrypted when the device is unlocked to prevent data leakage if a device is lost, stolen, or re-used. Keys required for data decryption when unlocking a device are based on the user's password.

Shared devices

Enterprises such as hospitals, banks, and airlines use shared devices for their employees. Knox supports shared devices so IT admins can manage device and security policies, and install applications with an EMM. Each employee can login separately with an Active Directory ID and password. For security and data privacy, user data is deleted when an employee logs out of their shared device.

Knox Active Protection (KAP)

If a device isn't managed by an EMM, end users can activate or deactivate *Knox Active Protection* (KAP) using the Smart Manager app. KAP uses both *Real-time Kernel Protection* (RKP) and DM Verity to provide integrity checking for system code and data. KAP is always on EMM-managed devices.

Android on a Samsung device

Android managed profiles benefit from key Knox security modules that protect the device its sensitive work data. Knox enables Android protection with the following Knox features:

- RKP actively prevents kernel code modifications
- PKM periodic kernal checks for code integrity
- DM-Verity to ensure application and data integrity on the partition
- Trusted Boot measures each software component during boot-time and securely stores the cryptographic hash of the next component in TrustZone memory before loading it
- Sensitive Data Protection APIs are available for apps in Managed Profiles. The native email app enables SDP once it's installed inside Managed Profiles.
- The TIMA and CCM TrustZone-based KeyStores provide storage for digital credentials such as VPN and email app certificates.
- Access to Managed Profiles depends on the integrity of the device. If the integrity check fails at the time of creating Android, it is not allowed. If an integrity check fails, the device cannot boot.

Android on a Samsung device does not require a Knox license activation fee. Knox security enhancements for existing Android managed profiles are updated seamlessly with *Over-the-Air* (OTA) updates.

Knox Enabled App (KEA)

Knox Enabled App is a per-app invisible container designed for application developers and vendors to provision services to device users. KEA allows service providers to deploy their applications and optimally use the Samsung Knox platform securely without the need for Enterprise Mobility Management (EMM). Since KEA is an invisible, unmanaged container, the user experience is the same as the original version of the application. Knox platform security extended to KEA provides end users data protection by encrypting app data. If a device is compromised, lost, or stolen, app data cannot be unencrypted.

The KEA workspace is implemented based on the Knox Workspace and customized according to use case requirements. Knox Workspace is created and managed by an EMM, and suitable for the enterprise environment. For individual app vendors and developers, creating, managing and configuring the KEA workspace presents challenges without an EMM. However, with KEA, the device automatically creates and manages the KEA workspace when the KEA app is installed.

Additional information (metadata) is required to operate as a KEA app. When a KEA app is installed in KEA-capable devices, the device detects the metadata and authenticates the app through a Knox License Manager (KLM) Server. Once authenticated, the KEA workspace is created and the app is installed inside the workspace, including the configuration of the *SE for Android Management Service* (SEAMS) container.

If the KEA app is installed in devices incapable of using KEA, including non-Samsung devices, the KEA metadata is ignored, and works like a regular Android app, which eliminates the need for a separate version of the app.

Virtual Private Network

The Knox platform offers additional comprehensive support for enterprise *Virtual Private Networks* (VPN). This support enables businesses to offer employees an optimized, secure path to corporate resources from their devices.

Knox offers the following VPN features for IPsec and SSL:

- Per-app connections
- On-demand connections
- Always-on connections
- Device-wide connections
- VPN chaining (nested connections)
- Blocking routes to prevent data leakage if a mandatory VPN connection drops
- Pushing VPN profiles to multiple managed devices
- Traffic usage tracking
- HTTP Proxy over VPN

Use Knox to configure VPN connections to enforce Web traffic redirection through an HTTP proxy server, allowing enterprises greater visibility into network traffic and device usage patterns of employees. The Knox VPN framework supports VPN configurations using a static proxy server IP and port, and web proxy authentication.

The Knox platform offers broad feature support for the IPSec protocol suite, including:

- Internet Key Exchange (IKE and IKEv2)
- IPsec IETF RFCs – IKEv1
- IKEv1 – Main and aggressive IKE exchange modes with pre-shared key, certificates, Hybrid RSA, and EAP-MD5 authentications
- IKEv2 with PSK and certificate-based authentication
- IKEv2 – Pre-shared key, certificates, EAP-MD5 EAP-MSCHAPv2 authentication methods, and mobile extensions
- Triple DES (56/168-bit), AES (128/256-bit) with MD5 or SHA
- IKEv1 Suite B Cryptography supported with PSK and ECDS signature-based authentications
- IKEv2 Suite B Cryptography supported with ECDSA signatures

The Knox supports leading SSL VPN vendors. Since SSL implementations are proprietary, Knox features a generic VPN framework which enables third-party SSL vendors to support their clients as plug-ins. Enterprise IT admins use Knox EMM policies to install and configure a SSL VPN client.

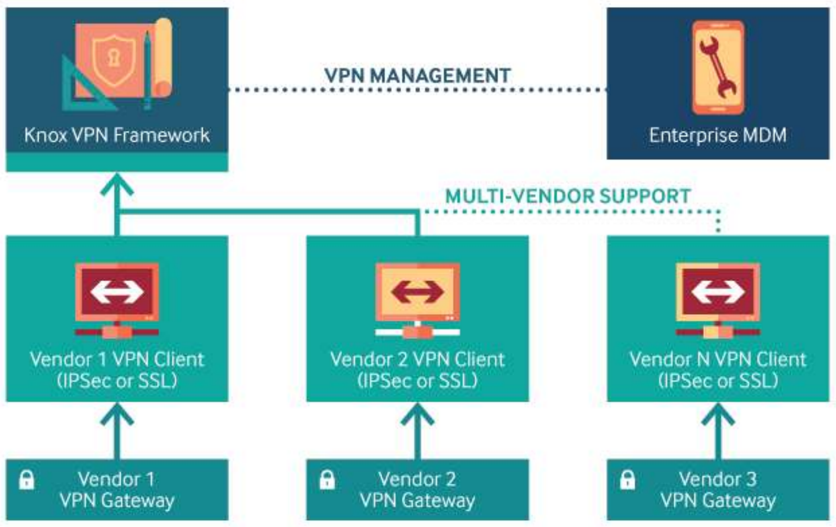

Figure 7 – Multi-Vendor Support in Knox

The per-application Knox Workspace VPN feature enables an enterprise to automatically enforce a VPN on just a specific set of applications. For example, an IT admin can configure an employee's device to enforce VPN for only business applications, ensuring data from the user's personal applications do not use the VPN and overload the company's Intranet resources. At the same time, user privacy is preserved because personal data does not enter the enterprise network.

Per-app VPN can also be applied to the Knox Workspace for some, or all, container applications.

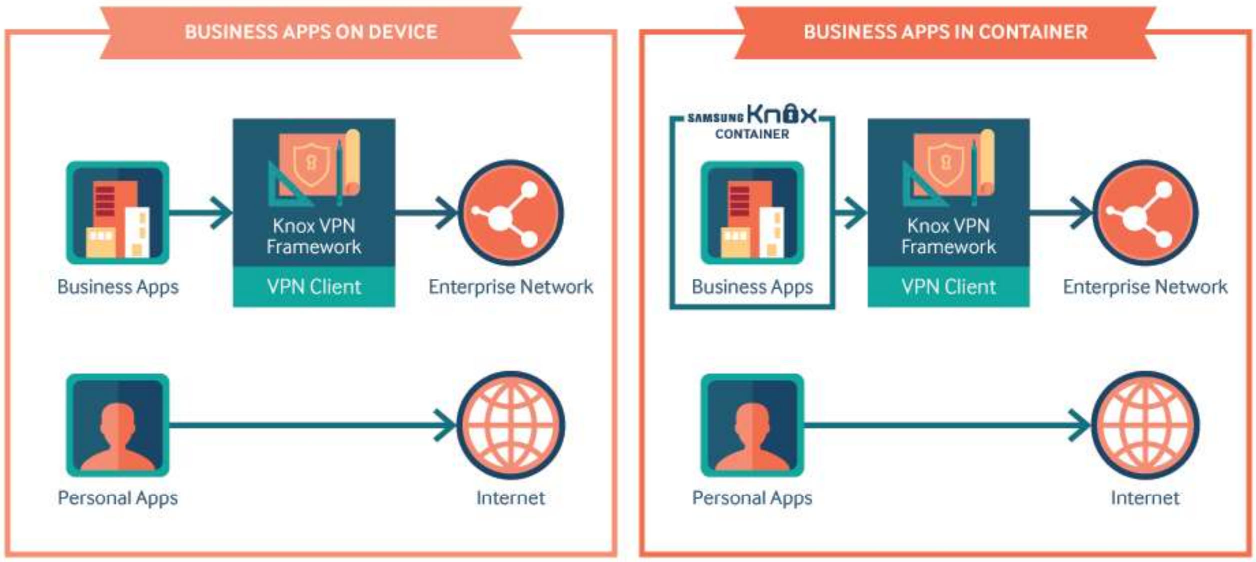

Figure 8 - Per-app VPN

Smartcard framework

The *United States Department of Defense* (US DoD) has mandated the use of Public Key Infrastructure (PKI) certificates for their employees to digitally sign documents, encrypt and decrypt e-mail messages, and utilize secure network connections. These certificates are typically stored on a Smartcard called the *Common Access Card* (CAC).

The Knox platform affords application access to the hardware certificates on the CAC via standards-based *Public Key Cryptography Standards* (PKCS) APIs. This type of access enables the use of the CAC card by the browser, e-mail application, and VPN client, as well as other custom government applications. Other enterprises show a growing interest to using Smartcards for the same purpose, especially those requiring robust security and information protection.

The Knox platform provides improved Smartcard compatibility via a software framework that allows third-party Smartcard and reader providers to install their solutions into the framework.

Active Directory integration

Knox provides an option to choose an Active Directory password as the unlock method for Knox Workspace. This has two important benefits. First, it allows IT admins to use a one-password management policy for desktop and mobile devices. Second, the end user only needs to remember one password to access all services offered by the employer, thereby reducing employee password fatigue and improving productivity.

At the heart of this feature is the proven industry-standard Kerberos protocol. Active Directory is the most widely-deployed enterprise grade directory service built-in support for Kerberos. Knox provides a set of workspace creation parameters to configure workspace to use the Active Directory password as the unlock method. Additionally, IT admins can also configure Single Sign-On for services inside workspace, along with the unlock method.

Enterprise Mobility Management

Knox provides 1,500 of Enterprise Mobility Management (EMM) security policies for fine-grained device control. The solution includes:

- *Enterprise Mobility Management* (EMM)
- *Mobile Application Management* (MAM)
- *Identity and Access Management* (IAM)

Knox EMM policies are designed to lower costs and improve device usability and manageability for small or medium sized enterprises. The full mobile and web application solution has cross-platform support for Samsung devices, other Android devices, and iOS^{TM} devices to support BYOD or COPE deployments. Support for cross-platform devices creates a centralized location for enterprises to manage devices. Mobile Application Management focuses on data management, as well as who has access to applications.

Identity and Access Management adds yet another layer of security with automated user authentication and easy access for administrators to monitor system activity.

Enterprises can use the cloud-based policy management, an on-premise Active Directory, or a hybrid combination to separate employees and external or partner users. The full mobile and web application solution has cross-platform support for Samsung devices, other Android devices, and iOS™ devices to support BYOD or COPE.

Knox API categories

Enterprise IT Compatibility

- Account Management using blacklisting/whitelisting
- Active Directory integration
- LDAP Management
- Enterprise Billing
- VPN

Security and Compliance

- Device Admin Management
- Firewall
- Password Management
- Device Security
- Remote Event Injection
- Audit Logging
- Usability
- Kiosk Mode
- Workspace Management
- Multi-user Mode

Device Control

- Date and Time
- Bluetooth
- Location Management
- Device Restrictions
- Wi-Fi Configurations
- APN Settings
- Device Inventory

Application Management

- Browser
- Email/Exchange Configuration
- Application Management

Telephony

- Telephony Management
- SIM Change Information
- Roaming Restrictions

Knox Mobile Enrollment (KME)

Enrolling an Android device into a company's EMM system typically begins with a user downloading the agent application from the Google Play store, then authenticating it. Enterprises are facing escalating help desk calls as more and more users are activating mobile devices for the workplace. When presented with prompts, privacy policies, and license agreements, users might experience difficulties during the process, resulting in a poor overall experience.

The Knox platform provides a simplified enrollment solution for supported EMMs that's streamlined, intuitive, and eliminates steps and human error potential.

Enrollment occurs using either self-discovery with an email domain, or employees are provided an enrollment link sent by email, text message, or through the company's internal or external website. Once the link is clicked and invoked, users are prompted to enter their corporate email address. This action triggers the display of all required privacy policies and agreements. After accepting the terms, users enter a corporate account password for enterprise authentication. Any agent application required is automatically downloaded and installed.

KME allows IT admins to enroll hundreds or thousands of employees at the same time. Samsung provides a web tool and an application to scan package bar codes (the device IMEI). KME is targeted for devices purchased for COPE enterprises and supported carriers and resellers.

Another option includes using a master device to automatically enroll devices using NFC. The master device is configured by downloading an app from the Playstore. Each device is enrolled in an EMM profile selected by the IT admin.

EMM vendors can utilize KME to simplify the onboarding process for their enterprise users, significantly improve the user experience, and reduce support costs.

KME supports multiple EMM configurations per account. With complex device environments, and multiple EMM profiles or configurations, KME enables IT admins to prepare hundreds of devices and connected them to the right EMM with ease. End users only need to turn on the device and connect to the network. KME takes care of activation without users needing to do a thing.

Enterprise Billing

Enterprise Billing provides a mechanism to separate enterprise data usage from personal data usage. This enables enterprises to compensate their employees for work expenditures, particularly in BYOD cases, or to only pay for work-related data in COPE cases.

The Knox platform supports Enterprise Billing on Knox version 2.2 and above, and requires EMM support.

Enterprises configure two *Access Point Name* (APN) gateways. One APN is for data associated with enterprise-approved apps, and a different APN is for all other personal data. Enterprises must first register with a network operator's enterprise billing service. Once a new APN is provisioned for business use, Knox Workspace can be enabled for that dedicated APN. IT admins can also select individual apps inside or outside workspace to use data over the enterprise APN.

Enterprise billing with a dedicated APN can:

- Separate data usage over the mobile Internet for legacy 2G/3G/4G connections
- Route data traffic from the workspace over the enterprise APN
- Provide the capability to select individual apps inside or outside the Knox Workspace to use data over the enterprise APN

The enterprise APN can also be configured to allow or deny roaming. When roaming is enabled, personal data is routed through the default APN, and enterprise data is routed through a dedicated enterprise APN. By default, roaming over the enterprise APN is disabled. When a user is roaming in a single Packet Data Protocol (PDP) network, all enterprise apps are automatically routed to the personal APN for work continuity.

If enterprise apps use a network VPN connection, the VPN profile can be configured to route data through the enterprise APN. Dual SIM devices can also be enabled for Knox Enterprise Billing. The primary, or first SIM slot, is automatically selected to configure an APN and activate Enterprise Billing on the device.

To avoid the personal use of a SIM card, IT admins can lock the SIM card with a unique PIN combination. This ensures the SIM can only be used for enterprise billing on the authorized device. In addition, dedicated enterprise APNs are restricted, and APN settings are not visible or editable on the device.

Users can check personal and enterprise data usage on a Knox device by navigating to the Settings menu. To view data usage, employees navigate to Settings > Data Usage > Mobile Tab (personal) or Enterprise Tab (work).

Endnotes

[1] Kaspersky Lab, “2016 saw 8.5 million mobile malweare attacks," February 28th, 2017. http://www.techrepublic.com/article/report-2016-saw-8-5-million-mobile-malware-attacks-ransomwareand-iot-threats-on-the-rise/

[2] Ibid

[3] Nielsen, “The Digital Consumer,” October 2013, p. 8. http://www.nielsen.com/content/dam/corporate/us/en/reports-downloads/2014%20Reports/the-digital-consumer-report-feb-2014.pdf

[4] Consumer Reports, “Smart phone thefts rose to 3.1 million last year, Consumer Reports finds,” May 2014. http://www.consumerreports.org/cro/news/2014/04/smart-phone-thefts-rose-to-3-1-million-last-year/index.htm

[5] FCC, “Report of Technological Advisory Council (TAC) Subcommittee on Mobile Device Theft Prevention (MDTP),” December 2014, p. 22. http://transition.fcc.gov/bureaus/oet/tac/tacdocs/meeting12414/TAC-MDTP-Report-v1.0-FINAL-TAC-version.pdf

[6] Workshare, “Data Guardian: Detecting Business Risk 2014," p. 14-16. https://d3liiczouvobl1.cloudfront.net/uploads/refinery/resource/file_name/251/Workshare_-_Data_Guardian_-_Detecting_Business_Risk_2014.pdf

About Samsung Electronics Co., Ltd.

Samsung Electronics Co., Ltd. is a global leader in technology, opening new possibilities for people everywhere. Through relentless innovation and discovery, we are transforming the worlds of televisions, smartphones, personal computers, printers, cameras, home appliances, LTE systems, medical devices, semiconductors and LED solutions. We employ 236,000 people across 79 countries with annual sales exceeding KRW 201 trillion. To discover more, please visit www.samsung.com

For more information about Samsung Knox, visit www.samsungknox.com

Samsung Electronics Co., Ltd.
416, Maetan 3-dong, Yeongtong-gu
Suwon-si, Gyeonggi-do 443-772, Korea

Version	Date
Samsung Knox Security Solution V2.2	May 2, 2017
Samsung Knox Security Solution V2.1	December 8, 2016
Samsung Knox Security Solution V2.0	Nov. 17, 2016
Samsung Knox Security Solution V1.12	September 22, 2016

Acronyms

AES	Advanced Encryption Standard
AOSP	Android Open Source Project
CAC	U.S. Common Access Card
CCM	Client Certificate Management
CESG	Communications and Electronic Security Group
CMK	Container Master Key
COBO	Corporate Owned Business Only
COPE	Corporate-Owned Personally Enabled
DAC	Discretionary Access Control
DAR	Data-at-Rest
DISA	U.S. Defense Information Systems Agency
DIT	Data-in-Transit
DRK	Device Root Key
DUHK	Device-Unique Hardware Key
FIPS	Federal Information Processing Standard
IAM	Identity and Access Management
IPC	Inter Process Communication
KEA	Knox Enabled App
MAC	Mandatory Access Control
MAM	Mobile Application Management
MCM	Mobile Container Management
MDM	Mobile Device Management
MMU	Memory Management Unit
NFC	Near Field Communication

Acronyms

NIST	National Institute of Standards and Technology
ODE	On-Device Encryption
PKCS	Public Key Cryptography Standards
PKM	Periodic Kernel Measurement
RKP	Real-time Kernel Protection
RP	Rollback Prevention
SBU	Sensitive But Unclassified
SDP	Sensitive Data Protection
SEAMS	SE for Android Manager Service
SE for Android	Security Enhancements for Android
SE Linux	Security Enhanced Linux
SRG	Security Requirements Guide
SSBK	Samsung Secure Boot Key
SSO	Single Sign-On
STIGs	Security Technical Implementation Guides
TIMA	TrustZone-based Integrity Measurement Architecture
VPN	Virtual Private Network